# INTELLIGENT PLATFORM MANAGEMENT INTERFACE

# PROTOCOL SECURITY

INTELLIGENT PLATFORM MANAGEMENT INTERFACE

PROTOCOL SECURITY

By

Syler W. Clayton

RECOMMENDED:

---

Dr. Brian Hay
Assistant Professor, Department of Computer Science
University of Alaska Fairbanks

---

Dr. Kara Nance
Professor, Department of Computer Science
University of Alaska Fairbanks

---

Dr. Jon Genetti
Associate Professor and Chair, Department of Computer Science
University of Alaska Fairbanks

---

Date

# INTELEGIENT PLATFORM MANAGEMENT INTERFACE PROTOCOL SECURITY

A

MASTER PROJECT

Presented to the Faculty

of the University of Alaska Fairbanks

in Partial Fulfillment of the Requirements

for the Degree of

MASTER of SCIENCE

By

Syler W. Clayton

Fairbanks, Alaska

April 2014

# Abstract

The Intelligent Platform Management Interface (IPMI) is a protocol that allows administrators to manage servers remotely. Hardware vendors including Dell, HP, Supermicro, IBM, Lenovo, Fujitsu and Oracle support IPMI though a Baseboard Management Controller (BMC) which can either be integrated into the motherboard or purchased as a pluggable module. The BMC runs silently alongside other components of the server and provides a lower level of hardware access than the Operating System (OS). This allows support for features like power cycling the server, mounting virtual media and accessing a remote console. The failure of BMC vendors to produce a more secure product, along with the inherent flaws of the IPMI protocol, increases the need for these systems' security capabilities to be evaluated. The IPMI protocol and various vendor implementations of the BMC has been the subject of recent scrutiny, and initial investigation has raised concerns about the security properties of these components. This project focuses on evaluating specific IPMI supported hardware and software setup in an environment modeled to simulate real use, for the explicit purpose of evaluating the security of the system. This project presents: several methods by which unprivileged users can gain remote access to the system, a list of best practices for proper configuration, a guide to clearing configuration settings before decommission, and a basic Metasploit module to scan for BMC related services.

# Acknowledgments

Research is often credited to one individual but cannot be completed without the helping hands of several. I would like to take this opportunity to acknowledge those whose encouragement, guidance, and patience, helped me complete this work.

I would like to thank my Mother for being my mentor, teacher and friend. My Father for igniting the passion for engineering, science and mathematics at a young age. My Fiancé, Angela, for putting up with my late nights of coding and always being there to support me.

I would like to also thank my advising committee. Thank you, Dr. Brian Hay, for presenting me with this research opportunity. Thank you, Dr. Jon Genetti, for giving me the opportunity to work for Dr. Hay on the Department of Defense, Code Scanner research project while my application for graduate school was being processed. I would also like to thank Dr. Kara Nance for helping create a solid security research program at UAF.

To my fellow colleagues, I am grateful for your attendance of my presentations and listening to my ideas. Specifically, I would like to thank Walker Wheeler for teaching me my first programming class (Java CS 103) and providing me with his project write-up to use as a template for this document. Brandon Marken, thank you for proof reading this paper. Victoria Avery, thank you for taking three thirty minute blocks at the Writing Center to help me edit this paper.

# Table of Contents

Page

# List of Figures

Page

# List of Tables

Page

# List of Source Code Files

Page

# *Introduction*

According to Dan Farmer, "the de-provisioning of old servers must be handled even more carefully, since eBay attacks and the like are a real threat," (Farmer, 2013). There is a growing need to analyze the security of Baseboard Management Controllers (BMCs) and their implementations. This document chronicles the security problems associated with the Intelligent Platform Management Interface (IPMI) protocol, the various vender implementations and user configuration.

The goal of this project is to provide best practices for user configuration and server decommission. We start out reviewing what BMCs are and why they are important for those systems to be secure. Then, we document how to set up a testing environment on a private network to configure Integrated Lights Out Interface (iLO) and Integrated Dell Remote Access Controller (iDRAC) devices. We then document how to recover network information from the previous setup and how to wipe data before server decommission. Finally we provide a Metasploit module to automate scanning for BMC related services for iLO and iDRAC.

# *IPMI Background*

In this section, we introduce the concepts of IPMI and the BMCs that implement the protocol. We also describe why IPMI security is a relevant issue.

## *BMC Diagram*

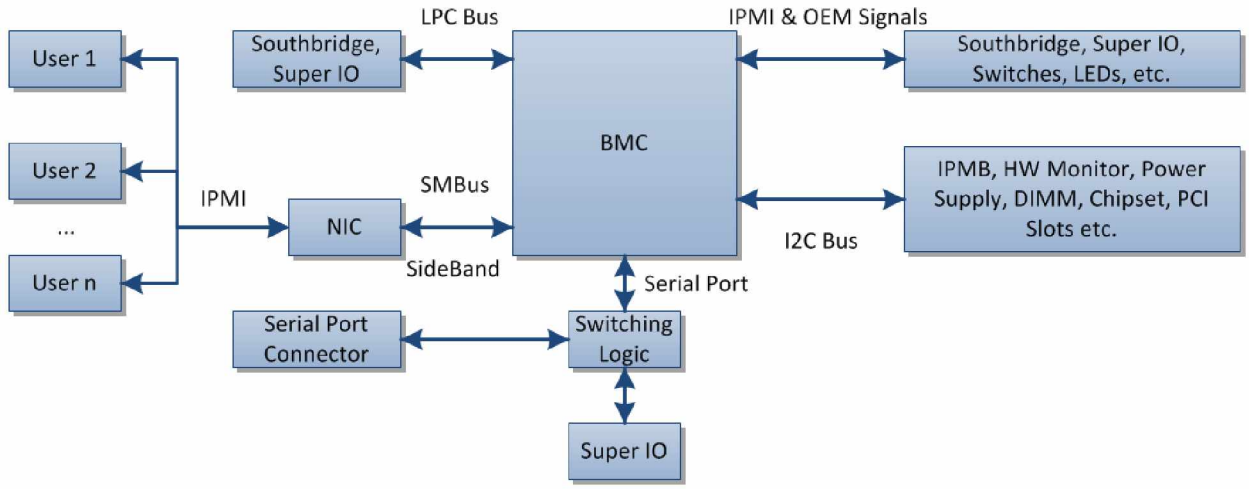



**Figure 1 – BMC Diagram**

## *What is a BMC?*

BMCs play a crucial role in server administration. They typically have their own Central Processing Unit (CPU), memory and storage. They are integrated into the sever motherboard or purchased as a pluggable module or Peripheral Component Interconnect (PCI) card. They are designed to monitor server physical health and operate in emergency situations (i.e. even when the server is powered off). They typically run Linux and support services such as the Simple Mail Transfer Protocol (SMTP), Lightweight Directory Access Protocol (LDAP), Secure Shell (SSH), and telnet. Some even allow mounting remote media, capturing snapshots, and executing remote commands on the server. Therefore, whoever controls the BMC, controls the server.

## *What is IPMI?*

The IPMI is a protocol for remote server management originally developed by Intel along with Cisco, Dell, HP and NEC in 1998 (Farmer, 2013). There are two specifications: version 1.5 (2001) and version 2.0 (2004) (Farmer, 2013). An embedded server called the Base Board Management Controller implements the IPMI protocol.

## *Why is Security an Issue in IPMI?*

IPMI is a powerful utility that allows system administrators to resolve issues remotely that would otherwise need direct physical interaction. Take, for example, receiving a call at 4 am in the morning stating that a few servers are unresponsive and need to be power cycled. Instead of having to crawl out of bed, get dressed and drive into work, the system administrator can login to the BMC and issue a power cycle remotely. Other notable power user utilities include remote access of the server and the ability to mount virtual media. However, with great power comes the great responsibility to ensure the security of the system, lest control fall into unauthorized hands. A malicious user that compromises the BMC has access privileges to the server that emulate a direct physical connection to the hardware. Imagine an invisible hacker running around in your data center with full physical access to all of your hardware–that is essentially what happens when a BMC is compromised. Since IPMI is supported by a myriad of hardware vendors, this leads to three classes of security vulnerability classifications: The IPMI protocol specification, the vendor implementation, and the configuration by end users. The goal of this project is to gain a better understanding of the vulnerabilities of BMCs and to determine best practices for securing them.

# *IPMI Security Threats*

Since IPMI is supported by a myriad of hardware vendors, this leads to three classes of security vulnerability: the IPMI protocol specification, the vendor implementation, and the configuration by end users. In this section we will discuss examples of security vulnerabilities for each of these categories.

## *IPMI Protocol*

The IPMI specification has several known flaws which can allow users to intercept clear text passwords, grab password hashes and grant system level access without any password (Cipher 0), depending on the IPMI version and configuration (Farmer, 2013). This is accomplished though communicating with IPMI over a Local Area Network (LAN) using the Remote Management Control Protocol (RMCP) on port 623. Hewlett-Packard (HP) ships their BMCs with a unique, random, eight digit password for their BMCs. However, Dell ships their BMCs with the default credentials, "root/calvin." Passwords are typically stored in plain text on the file system and the flash memory stores backups of previous firmware, which could lead to even further password recovery in relation to the history of the server (Farmer, 2013). In our study, we were not able to gain root access to the iLO 2 or iDRAC 4 file systems (as vendors generally lock these types of privileges). However, in our research, we found a method to gain root access to the iDRAC 5 file system by locating a reference to "vendor mode" in the iDRAC 5 source code and then issuing hidden Remote Access Controller Admin (racadm) commands to put the card in vendor mode (Nielsen, 2014).

## *Vendor Implementation*

Each vendor packages their IPMI implementation with a set of preconfigured services which may lead to vulnerabilities if they are not kept up to date. The update process is typically done through the web based interface which is similar to that of a router. Upgrading IPMI related hardware does not require a server reboot, as BMCs are their own segregated system. Depending on the vendor, BMC firmware is either publically available or only available to users with a service contract. IBM and Oracle for example, both require service contracts (Farmer, 2013). Vendors typically abandon providing firmware and security updates to old BMC hardware. For example, Dell releases a new iDRAC hardware revision every 1 to 5 years; the current hardware iteration for iDRAC being the iDRAC 7. Once a new BMC hardware specification is out, the vendor will typically phase out providing support and firmware updates for older models. Since BMCs typically run Linux, they fall under the GNU General Public License (GPL) license. Intel has made their source code available for their BMCs. However, this does not mean they have to support installing custom firmware on their system directly.

An implication of lack of transparency is that it would be very hard to remove a hacker whom had gained persistent access to the BMC. Since there is very little information available from BMC vendors about the hardware architecture, it is possible for an attacker to lock a legitimate user out forever (short of taking a hammer to the BMC) (Farmer, 2013). A recent example of a vendor implementation issue is the recent vulnerabilities in Supermicro's IPMI implementation that targeted buffer overflow attacks in the HTTP web interface, as well as a URL directory navigation attack due to lack of input sanitization (i.e. grab any file from the file system) (Bonkoski, 2013).

## *User Configuration*

As noted in the "Related Work" section, 60% of the IPMI 2.0 servers scanned, reported that Cipher 0 was enabled which allows for passwordless authentication (Farmer, 2013). This is an example of users taking the default configuration for granted, without assessing the security implications of the related services and features directly. All of the IPMI servers that were scanned were assigned public IP addresses. Users should be segregating IPMI supported devices on a private network and protecting them with a firewall. Lastly, there is the issue of minimum password length and complexity. If the user does not have a strong password policy in place, it could lead to a compromise of the hardware through brute force attack methods.

## *Related Work*

Dan Farmer, a security researcher in the DARPA Fast Track Program, published version 1 of his groundbreaking IPMI research at the beginning of 2013 and stated, "there's a desperate need for security research, articles, software tools, and discussion about the implications of IPMI and all it entails," (Farmer, 2013). Farmer gathered data regarding IPMI related services by sending a, "Get Channel Authentication Capability," packet to all IP addresses on the Internet in a 24 hour timeframe. Farmer noted, "312,357 IP addresses responded to the initial scan, with 295,364 replying with a valid IPMI response packet," (Farmer, 2013). Of the replies, 2/3rds indicated that only IPMI version 1.5 was supported, with no cryptographic protection other than optional password hashing, (Farmer, 2013). Of the servers that supported IPMI version 2.0, it was determined that 57.8% had Cipher 0 enabled, which allows passwordless authentication (Farmer, 2013). There was a response in July from H.D. Moore, whom put together a set of methods to exploit various vulnerabilities in IPMI with Metasploit, including Cipher 0 authentication and password hash dumping via the IPMI 2.0 RAKP Authentication Remote Password Hash Retrieval Vulnerability (Moore, 2013). H.D. More is the developer of the Metasploit framework, a penetration testing software suite, and the founder of the Metasploit Project. He is currently Chief Research Officer at Rapid7 (Moore, 2013). In August of 2013 a paper titled, "Illuminating the Security Issues Surrounding Lights-Out Server Management," was published and included a proof of concept buffer overflow exploit for Supermicro devices (Bonkoski, 2013). On November 6th, H.D. Moore published "Supermicro IPMI Firmware Vulnerabilities," which disclosed buffer overflow vulnerabilities in Supermicro's IPMI web interface (CVE-2013-3621, CVE-2013-3623) (Moore, 2013). Moore also disclosed a vulnerability targeting url_redirect.cgi which allows a "directory traversal attack due to lack of sanitization," which would allow an attacker access to any content on the file system (Moore, 2013). On November 15th Juan Vazquez, a security researcher for Raid7, published a functional exploit in response to H.D. Moore's CVE-2013-3621 vulnerability (Vazquez, 2013).

## *Project Scope*

The IPMI protocol and the BMCs that support it provide an enormous set of investigation opportunities. To make the analysis of IPMI supported systems more manageable for this research effort, the following scope limitations were imposed on this project:

1) Set up a test bed for IPMI related hardware on a private network.
2) Document how to configure iLO to harden against security vulnerabilities.
3) Document how to wipe data before disposing the server.
4) Build a module in Metasploit to automate testing for services related to the HP IPMI implementation known as the Integrated Lights Out Interface (iLO).
5) Look at iDRAC (Dell's implementation) if there was time (i.e. stretch goal).

## *Lab Setup & Configuration*

It is imperative to ensure that BMCs are only exposed on a segregated, private network to avoid unwanted attention. In this section, we will discuss setting up a private network to create a test bed for IPMI supported hardware. We gathered four HP servers and one Dell server which had been previously decommissioned from UAFs surplus center. We installed Ubuntu 12.04 LTS on one of the HP servers with Metasploit and ipmitools. We then set up a testing environment with the remaining three HP servers and one Dell server on our private network.

## *Network Configuration*

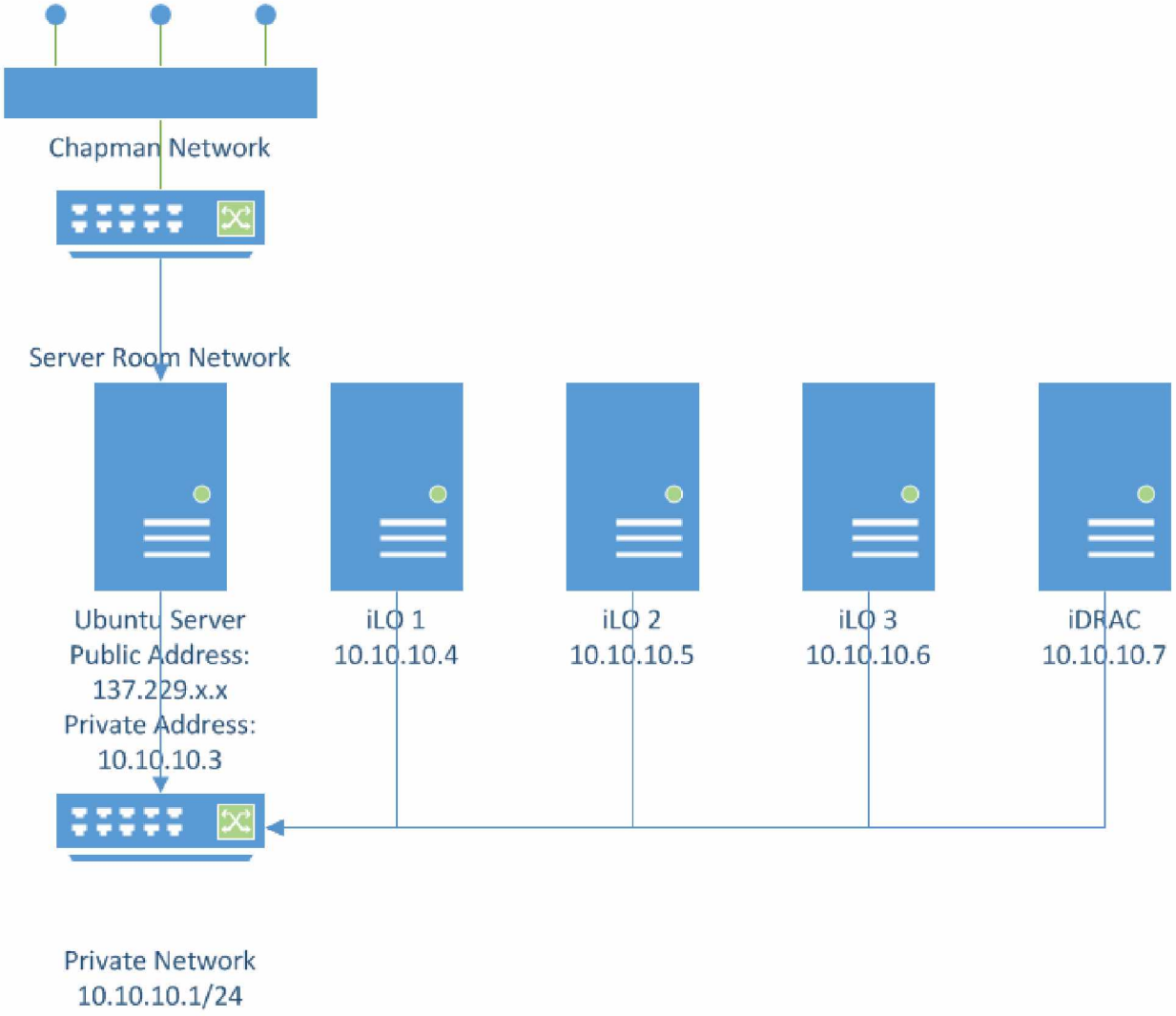
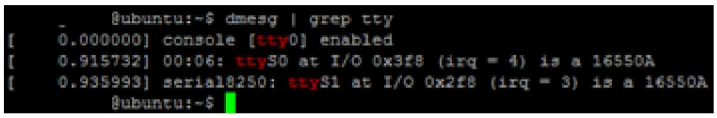



**Figure 2 - Network Configuration**

## *Configuring a Cisco Switch*

1) Find the serial port by running the command:

   ```
   desmg |grep tty
   ```



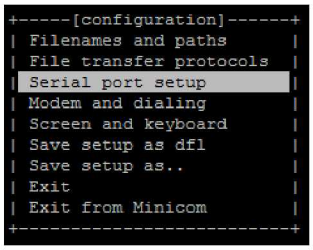

2) Therefore, we will be connecting on /dev/ttyS0

3) The program we will be using to connect to the serial port is called Minicom. First we must install the package by running the command:

   ```
   sudo apt-get install minicom
   ```

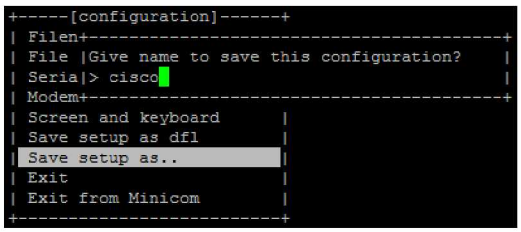
4) Next, we run the following command to configure the setup:

   ```
   sudo minicom -s
   ```



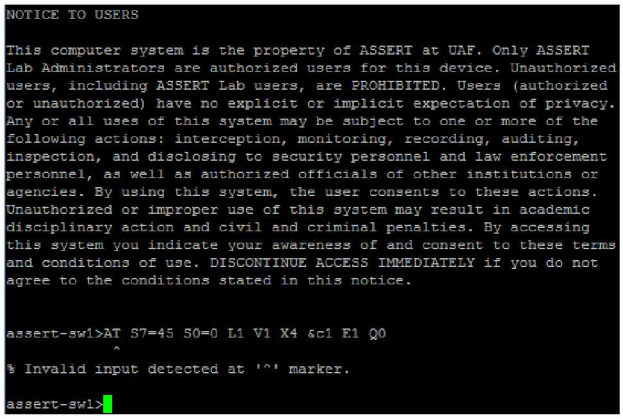

5) Now we select "Serial port setup" and specify these settings:

6) Finally we save the configuration file:




7) To launch a Minicom session using our configuration we type the command:

```
sudo minicom cisco
```




8) Now we're connected to the switch! This switch was already preconfigured.

## *Setting up the Network*

On one of the HP servers we installed Ubuntu 12.04 LTS to access the BMCs remotely, run ipmitools, and the Metasploit framework. eth0 is the public network on Figure 2. eth1 is the switch that we set up as a private network for connecting our BMCs on Figure 2.

1) First run the command:

```
sudo nano /etc/network/interfaces
```

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 199.165.X.X
netmask 255.255.X.X
network 199.165.X.X
gateway 199.165.X.X

auto eth1
iface eth1 inet static

address 10.10.10.3
netmask 255.255.255.0
network 10.10.10.0
broadcast 10.10.10.255
```

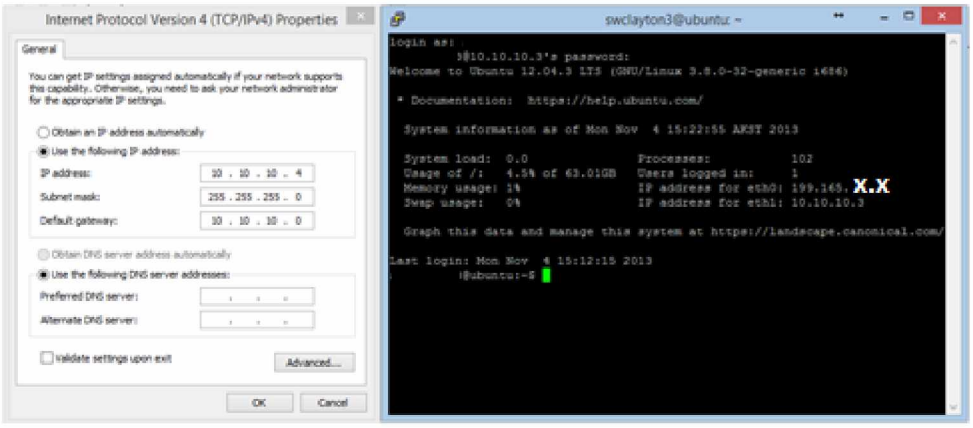
2) Now we run the command:

```
sudo /etc/init.d/networking restart
```

3) Finally we run the command `ifconfig` to verify that our network is configured properly:

```
eth0      Link encap:Ethernet  HWaddr 00:15:60:0c:X:X
          inet addr:199.165.X.X  Bcast:199.165.X.X  Mask:255.255.X.X
          inet6 addr: fe80::215:60ff:X:X/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:36 errors:0 dropped:0 overruns:0 frame:0
          TX packets:31 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3362 (3.3 KB)  TX bytes:3994 (3.9 KB)
          Interrupt:25

eth1      Link encap:Ethernet  HWaddr 00:15:60:0c:78:09
          inet addr:10.10.10.3  Bcast:10.10.10.255  Mask:255.255.255.0
          inet6 addr: fe80::215:60ff:fe0c:7809/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:250 (250.0 B)
          Interrupt:26
```

4) To test the configuration connected, we connected directly to the switch and SSH'd into the Ubuntu box:



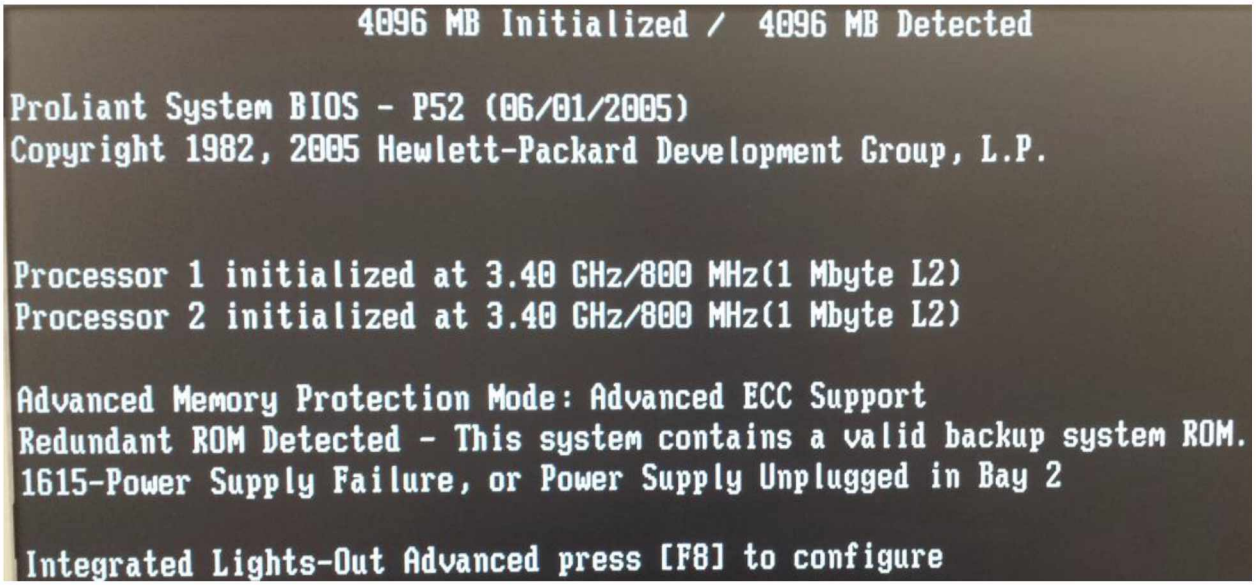
# *Integrated Lights Out Interface (iLO)*

In this section we will discuss: iLO configuration, an example of misconfiguration, data recovery, and proper methods for wiping data before server decommission.

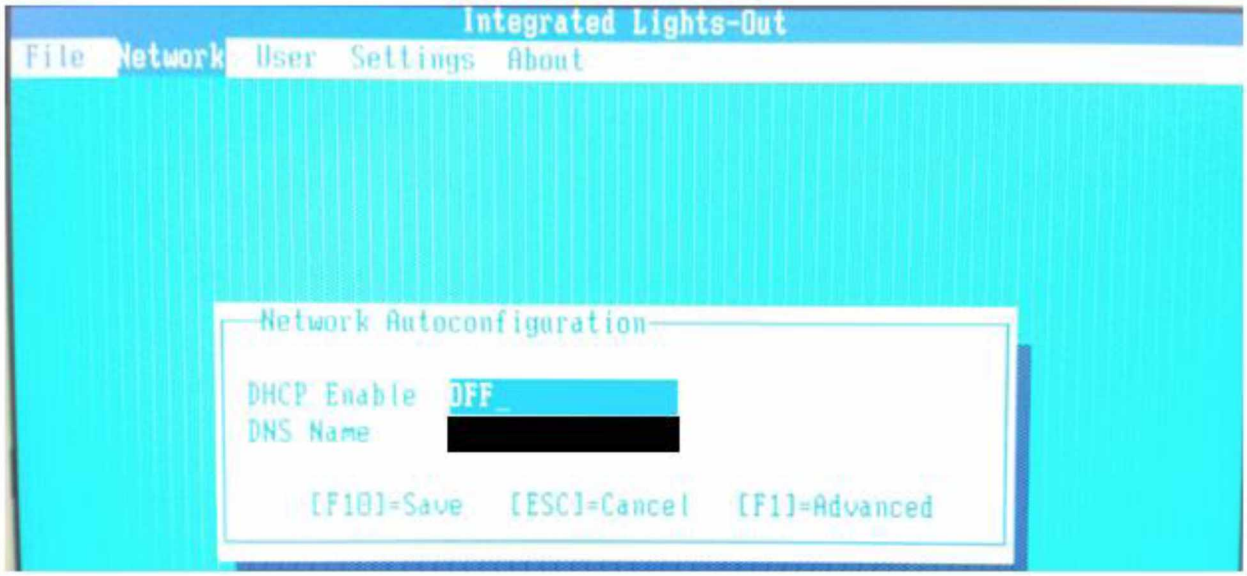
## *iLO Configuration*

Now that we have our private network setup, it's time to configure iLO to be able to connect via LAN.

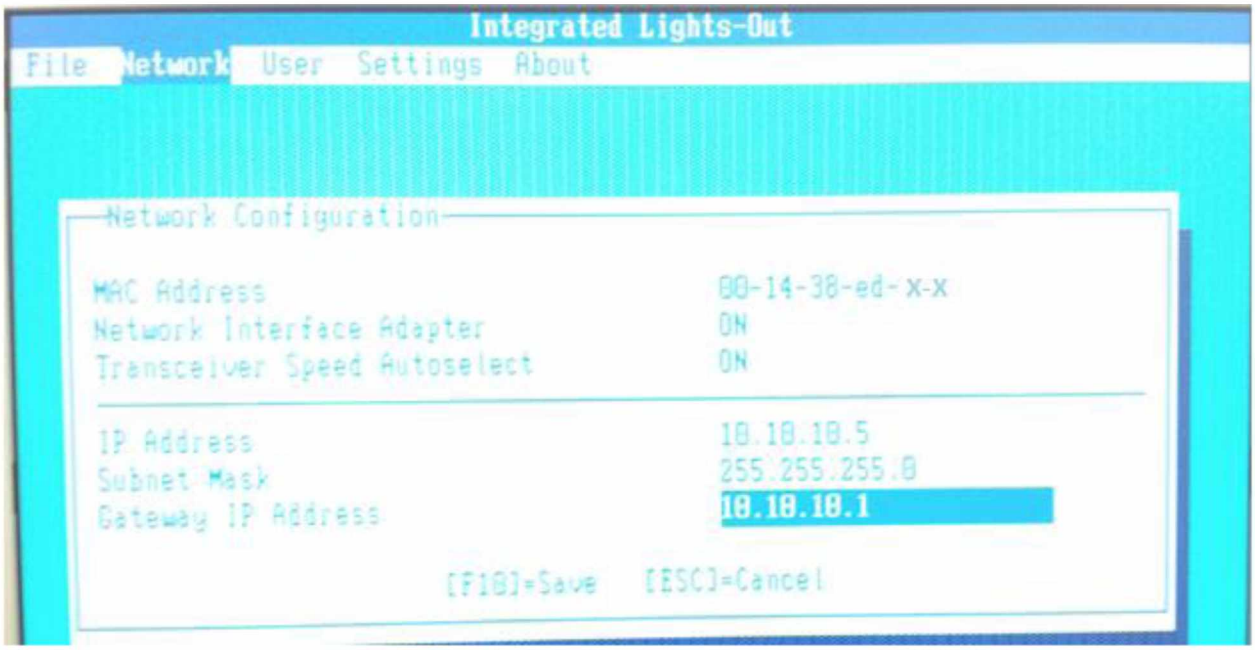
1) On POST press F8 to enter the Integrated Lights-Out Advanced configuration utility:



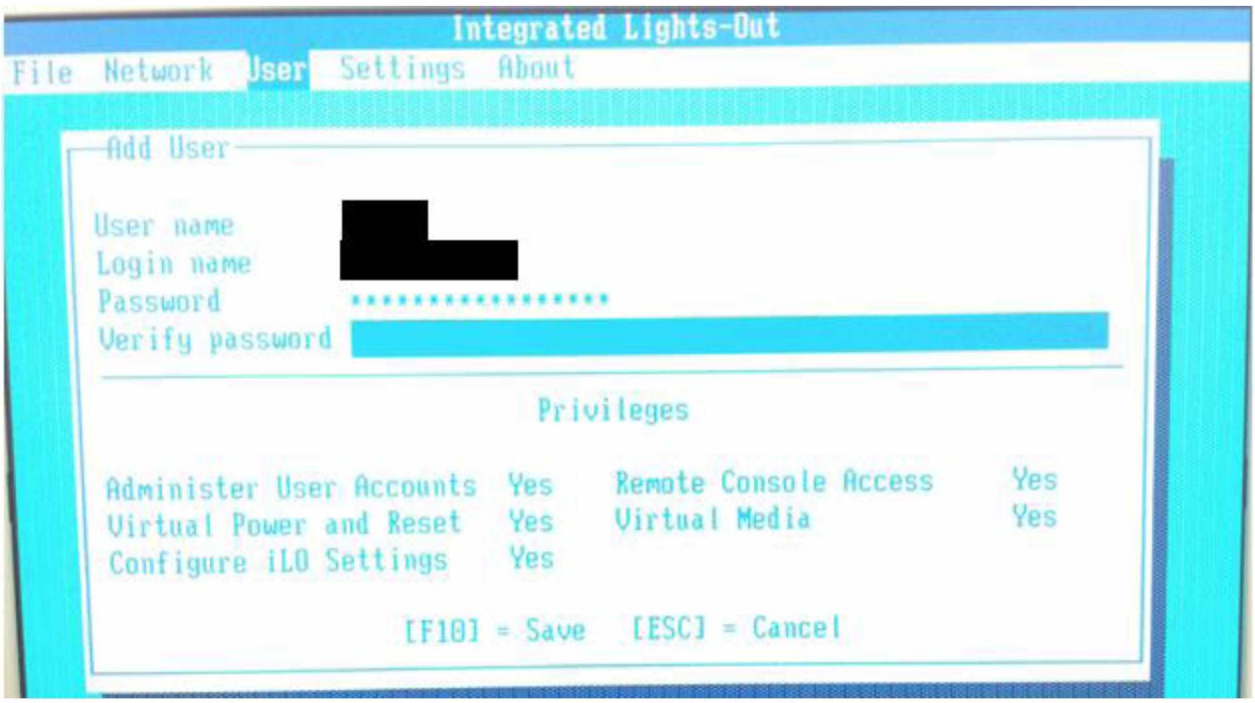
2) Next, go to Network->Network Autoconfiguration and make sure that "DHCP Enable" is set to "OFF":




3) Now, go to Network->Network Configuration and enter in the appropriate network information:



4) Finally, go to User->Add and create a new user:



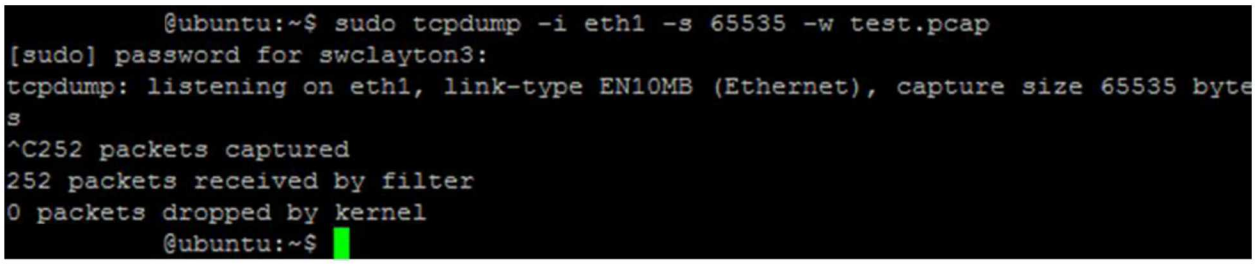

5) The destination should now be reachable:

```
          @ubuntu:~$ ping 10.10.10.5
PING 10.10.10.5 (10.10.10.5) 56(84) bytes of data.
64 bytes from 10.10.10.5: icmp_req=1 ttl=64 time=0.859 ms
64 bytes from 10.10.10.5: icmp_req=2 ttl=64 time=0.478 ms
64 bytes from 10.10.10.5: icmp_req=3 ttl=64 time=0.485 ms
64 bytes from 10.10.10.5: icmp_req=4 ttl=64 time=0.482 ms
64 bytes from 10.10.10.5: icmp_req=5 ttl=64 time=0.494 ms
64 bytes from 10.10.10.5: icmp_req=6 ttl=64 time=0.487 ms
64 bytes from 10.10.10.5: icmp_req=7 ttl=64 time=0.485 ms
64 bytes from 10.10.10.5: icmp_req=8 ttl=64 time=0.483 ms
^C
--- 10.10.10.5 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7000ms
rtt min/avg/max/mdev = 0.478/0.531/0.859/0.126 ms
          @ubuntu:~$
```

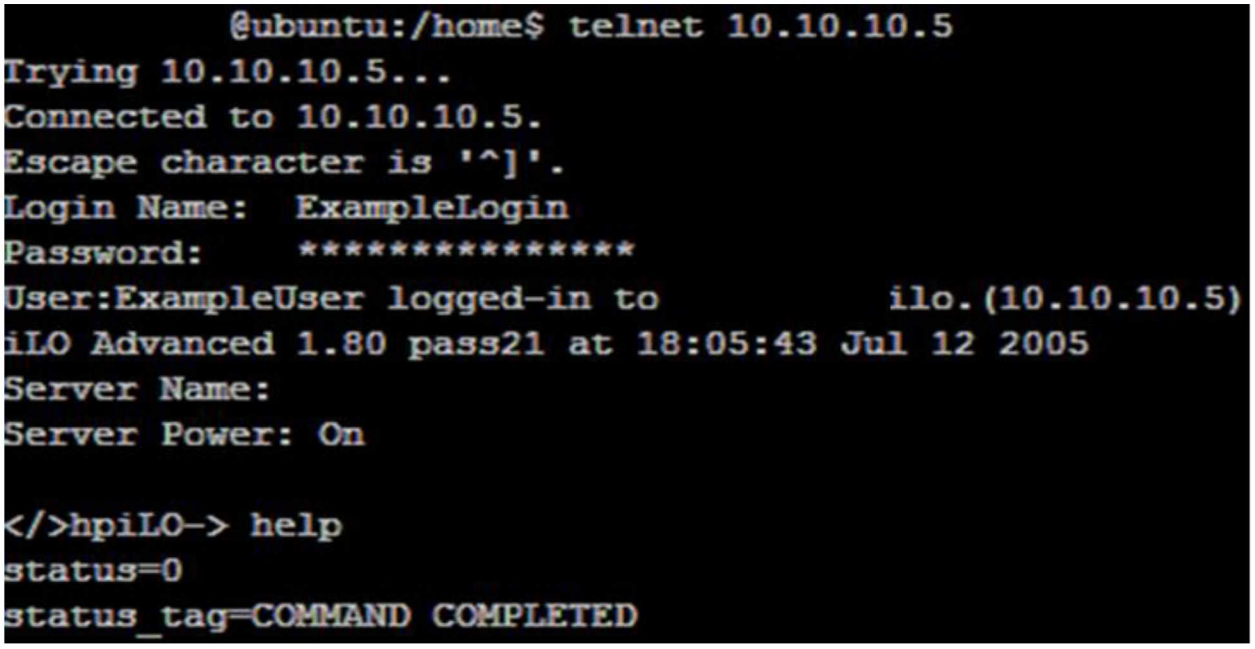
## *Example of Misconfiguration*

Telnet is enabled on port 23. We were able to use tcpdump to sniff passwords and communication in plaintext during connection.

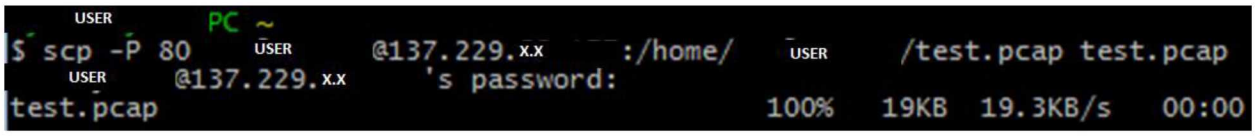
1) Start tcpdump capture:



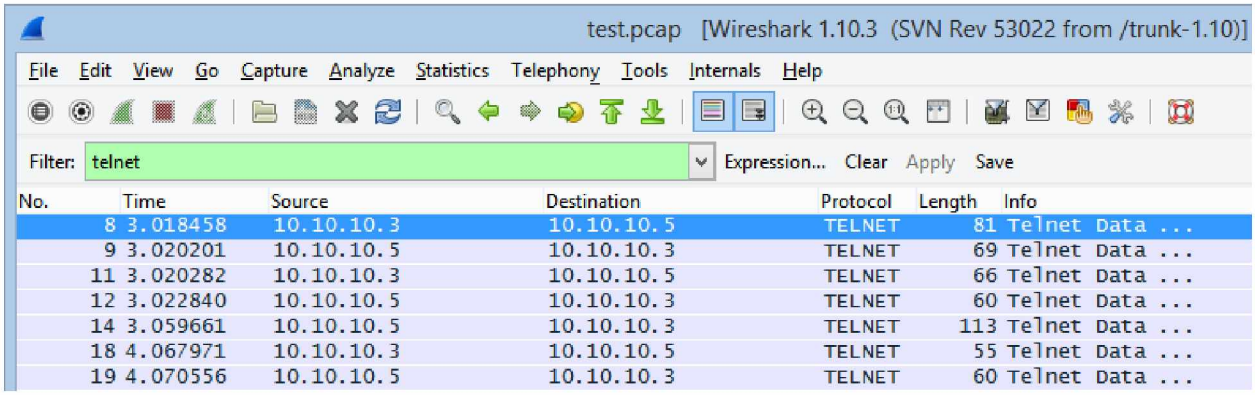

2) Telnet into BMC:



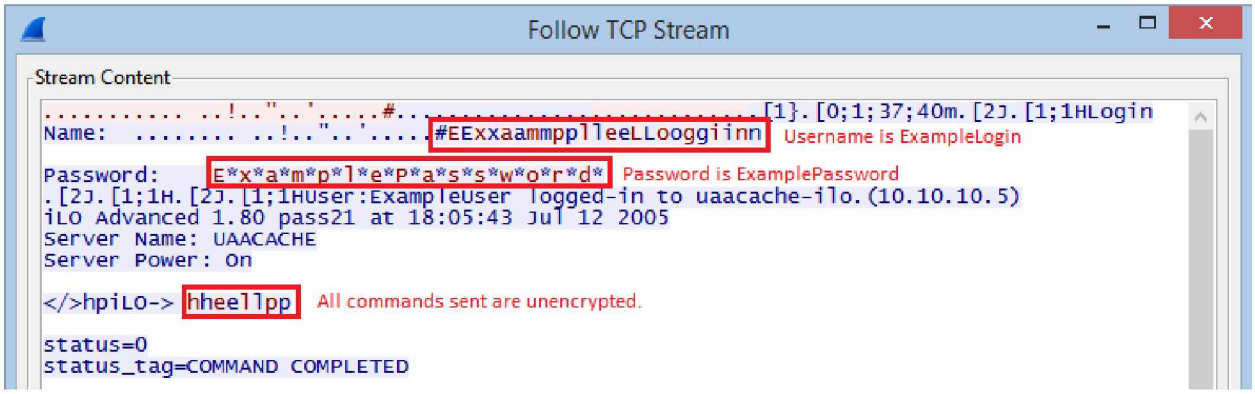

3) Scp the pcap file over to Windows box:



4) Open the pcap in Wireshark:




5) Right click on the first packet and click "Follow TCP Stream":

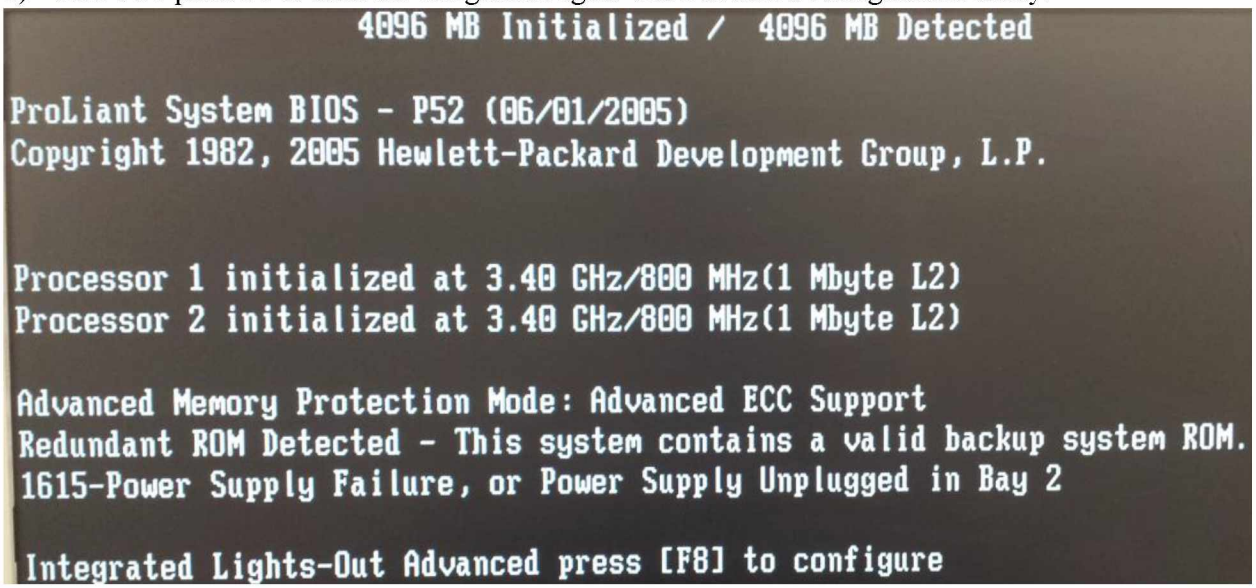



6) This can be fixed by going to Administration->Global Settings, disabling "Remote Console Port Configuration," and ensuring "Remote Console Data Encryption" is enabled.

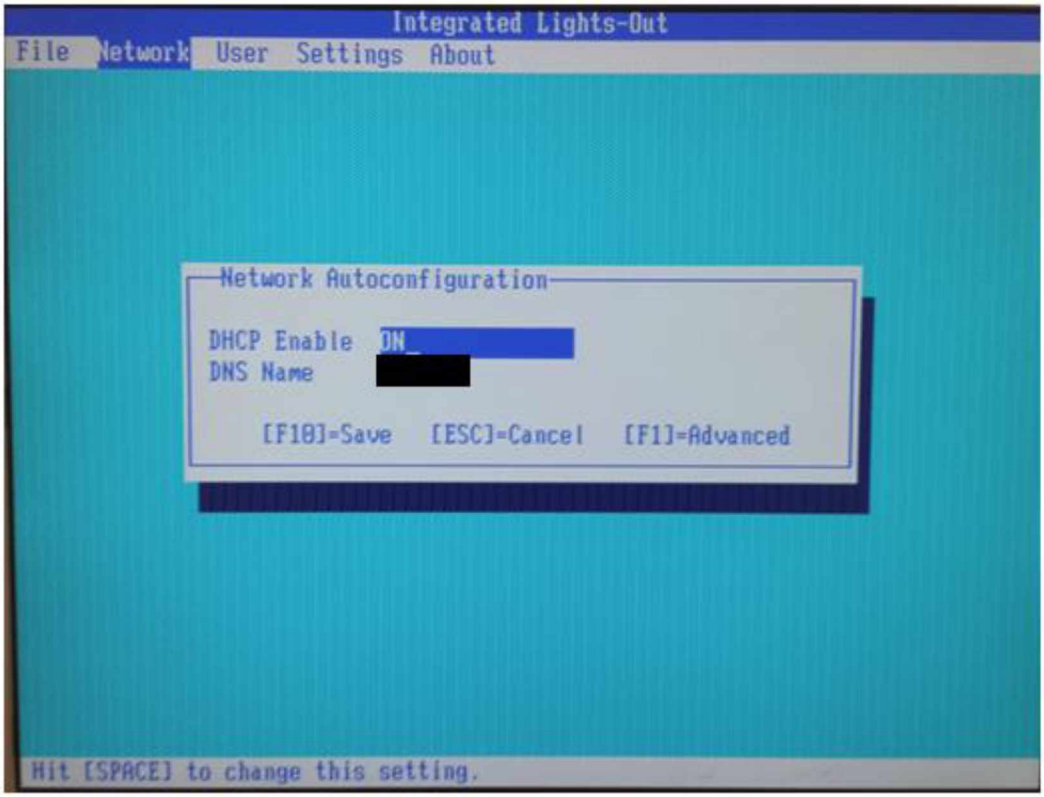
## *Recovering Data*

Information related to the network a server was hosted on, can be found in the BMC settings at startup. From the servers we picked up at surplus we were able to grab network related information from the iLO settings. This could allow an attacker to gain information related to the current network infrastructure of the environment it was previously decommissioned from. This could have been avoided if the previous owners had gone into the settings and restored the factory default configuration. BMCs passwords are typically universal in an enterprise environment and are updated rarely due to the inconvenience of password management. An attacker could dump passwords from a previously decommissioned server with the IPMI 2.0 RAKP Authentication Remote Password Hash Retrieval Vulnerability or by developing an exploit to gain root access to the file system. The iLO servers discussed in the next few sections do not support the IPMI 2.0 and there is no known exploit for gaining root access to the file system. However, the passwords are sitting there on the iLO file system, waiting to be uncovered. The file system resides onboard the integrated BMC. Thus, the only way to completely thwart the most determined hackers is to physically take a hammer to the board before decommission. It is truly saddening to see such a primitive solution that destroys the functionality of the board which could be avoided by vendors supporting custom firmware, as well as providing transparency and documentation with their BMCs.

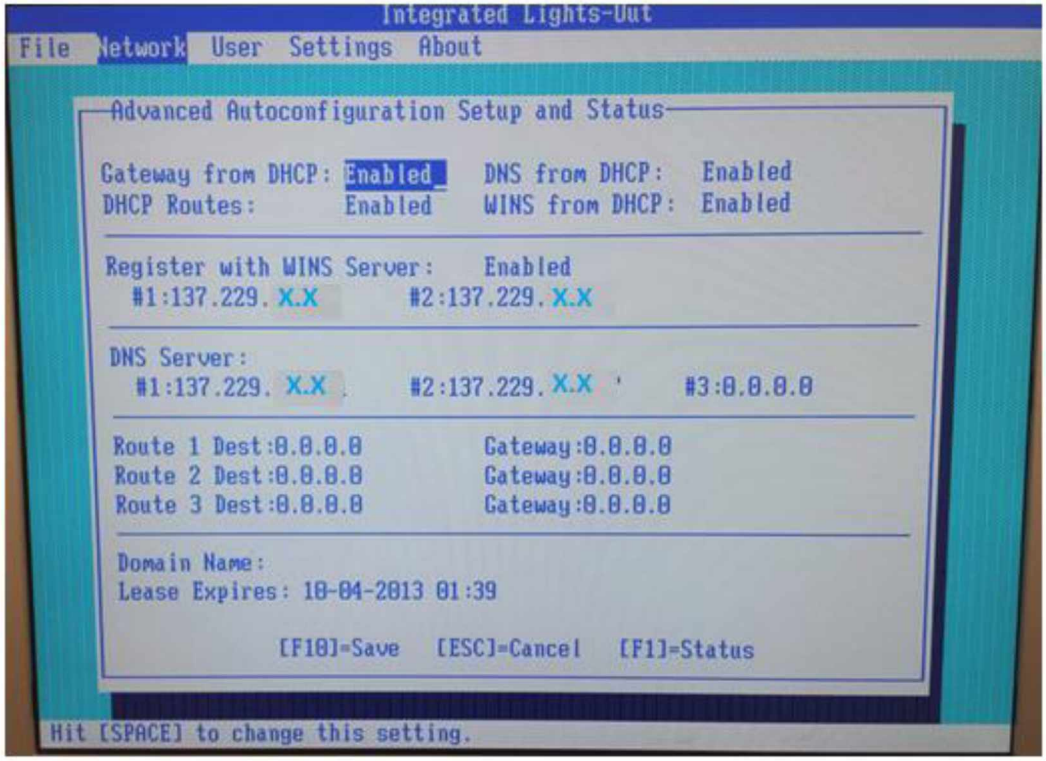
1) On POST press F8 to enter the Integrated Lights-Out Advanced configuration utility:



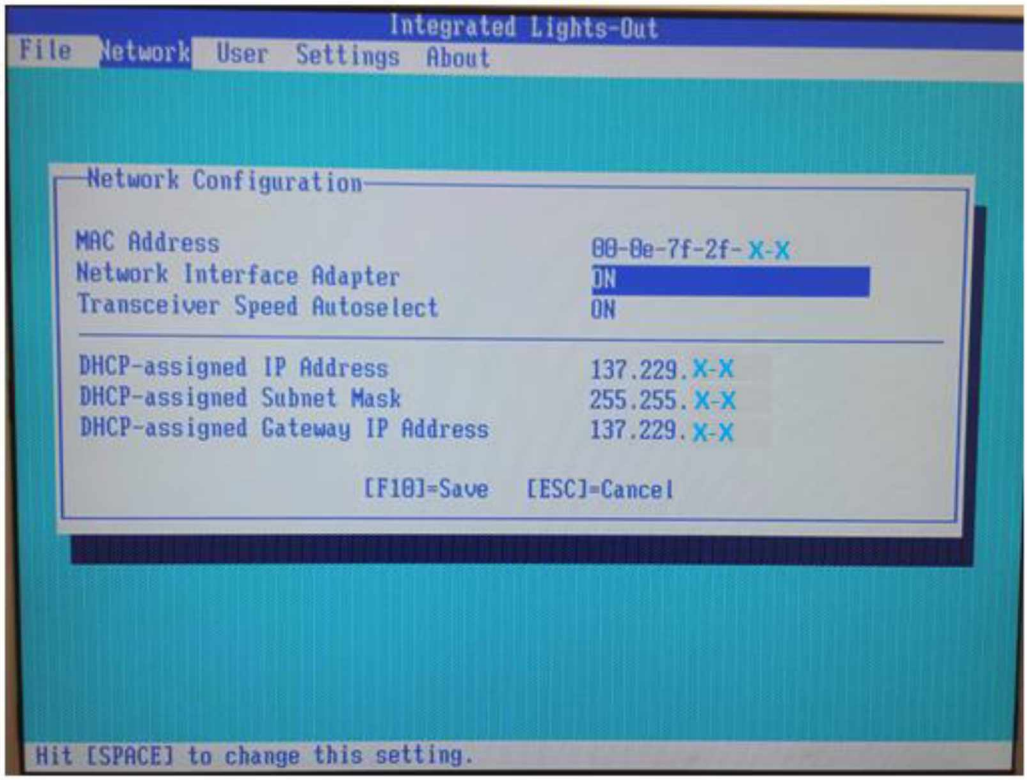

2) Next, go to Network->Network Autoconfiguration to see the DNS name:



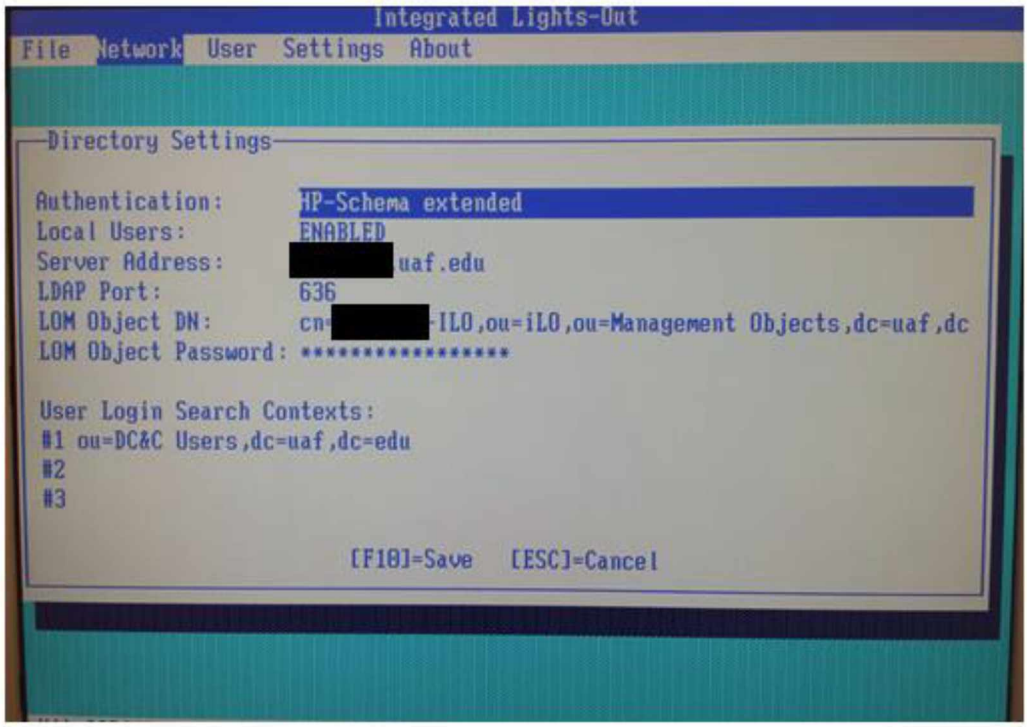
3) Press F1 to gather more detailed information:




4) Now, go to Network->Network Configuration:



5) Finally, go to Network->Directory:



## *Server Decommission*

The iLO web based interface allows users to clear some of the BMC settings in preparation for server decommission. However, as discussed in the previous section, it should be noted that this information is still typically floating around on the BMC's file system. As with most firmware upgrade processes, BMC vendors typically store several backups of the file system from previous firmware upgrades. This could allow a hacker to potentially compromise multiple previous owners of the hardware. Vendors don't allow root access to the file system but, this hasn't stopped researchers from gaining access on specific vendor implementations (Nielsen, 2014). Therefore, until vendors provide transparency of their BMCs, and support to remove such files, the only way to truly ensure no trace of data is left behind is to physically destroy the BMC. Below are instructions to clear easily accessible information:

1) Under "System Status" navigate to "iLO Event Log," scroll down and click "Clear Event Log." The iLO Event Log keeps track of power, clock settings, server resets, user logins, etc.
2) Under "System Status" navigate "iLO Event Log," scroll down and click "Clear IML." The iLO Management Log (IML) keeps track of warning and error messages.
3) Under "Administration" navigate to "User Administration" and delete each user. The User Administration section keeps track of users and their associated privileges.
4) Under "Administration" navigate to "Network Settings" and clear the "IP Address," "Subnet Mask" and "Gateway IP Address" fields. Clear the fields under "Advanced Configuration Parameters" and click "Apply."
5) Under "Administration" navigate to "SNMP/Insight Manager Settings," clear all fields and click "Reset Settings." The SNMP/Insight Manager Settings configures the SNMP Alerts server destination and configuration.
6) Under "Administration" navigate to "Directory Settings," clear all fields and click "Apply Settings." The Directory Settings section allows configuration of directory server authentication.

7) Under “Administration” navigate to “SSH Key Authorization” and remove any “Authorized SSH Keys.” The SSH Key Authorization section allows setting up authorized SSH keys for an SSH session.
8) Under “Administration” navigate to “Two-Factor Authentication” and click “Delete this Trusted CA Certificate” if applicable. The Two-Factor Authentication section allows setting a Trusted CA Certificate.

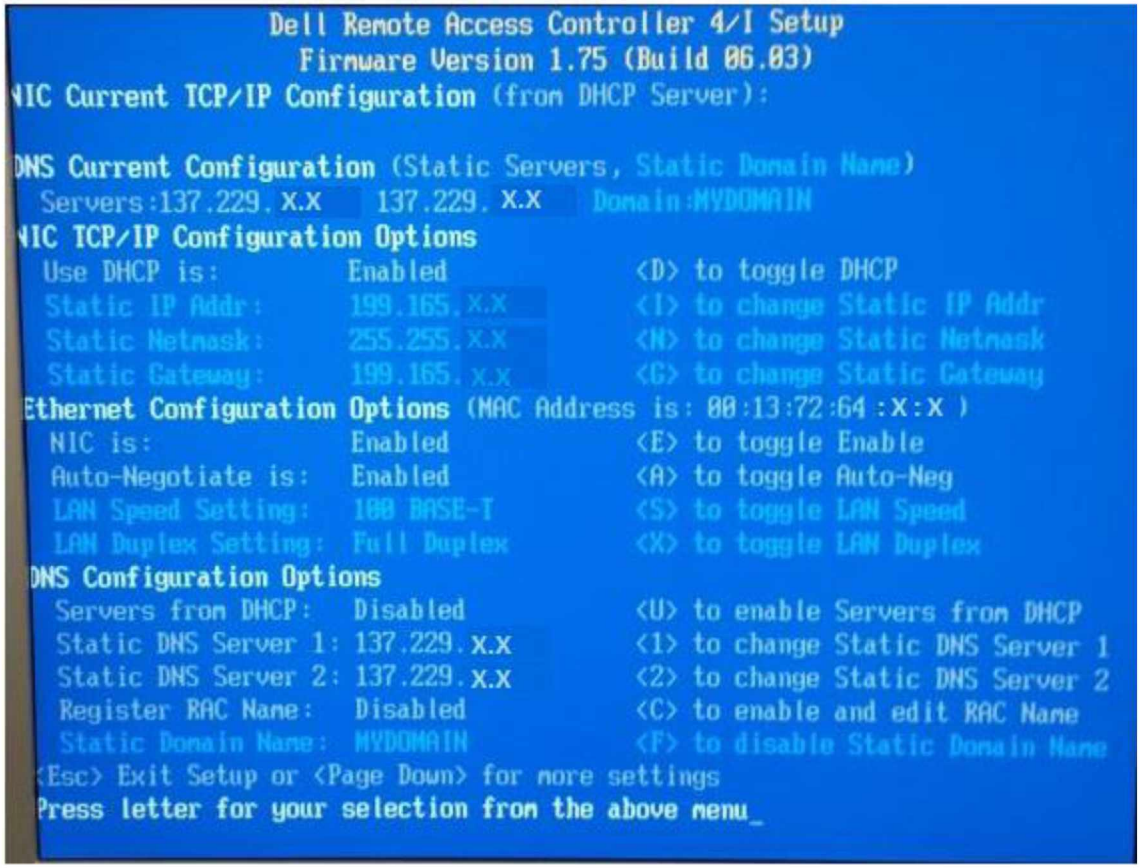
# *Integrated Dell Remote Access Controller (iDRAC)*

In this section we will discuss: iDRAC configuration, iDRAC IPMI over LAN setup, data recovery and proper methods for wiping data before server decommission.

## *iDRAC Configuration*

Next we will look at how to set up Dell's iDRAC.

1) On POST press CTRL+D to enter the Dell Remote Access Controller Setup




2) Press "E" to enable NIC under "Ethernet Configuration Options."

3) Next, enter the appropriate network information.

### *IPMI Over LAN*

IPMI Over LAN allows users to configure the BMC directly via the IPMI 1.5 or 2.0 protocol. The iLO servers that were decommissioned did not support IPMI over LAN. The iDRAC servers supported the 1.5 protocol. The IPMI 1.5 protocol only supports optional password hashing. This means that IPMI 1.5 is "vulnerable to a plethora of network security attacks such as password sniffing, network spoofing, connection hijacking, Man-in-the-Middle attacks, and more," (Farmer, 2013). The main vulnerabilities in the IPMI protocol do not exist in version 1.5 as the RAKP Authentication Remote Password Hash Retrieval and Cipher 0 vulnerabilities were introduced in version 2.0. iDRAC servers are natively configured to salt the password before using a MD5 hashing algorithm. IPMI, "uses a dynamic hash for an authentication code to validate a request," (Farmer, 2013).

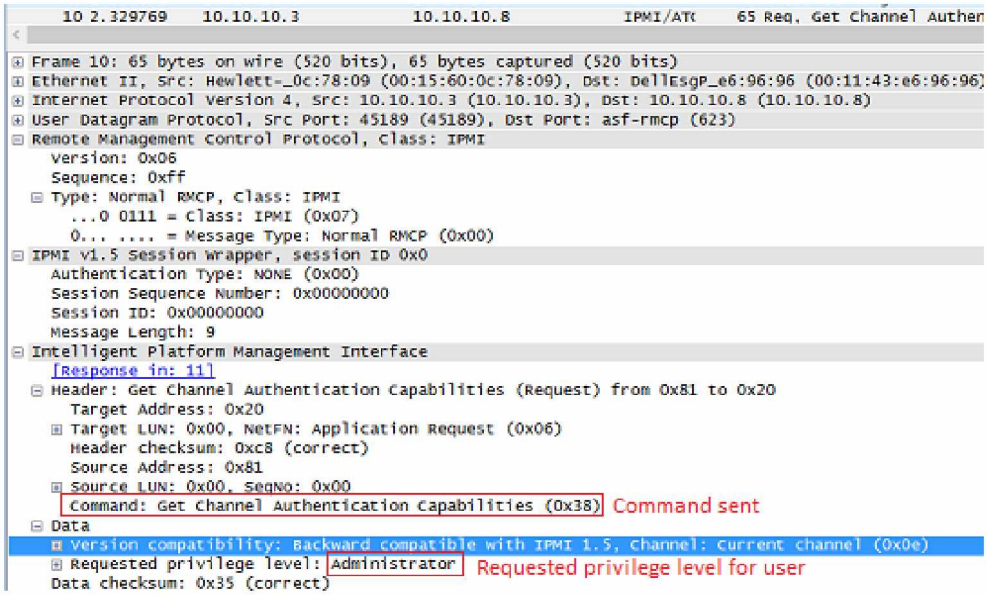
$$AuthCode = MD5(password + Temp\ Session\ ID + challenge\ string + password).$$

"The challenge string and session ID are determined by the remote client and the managed BMC prior to this request, and may change with any session request, and without the clear text password at hand this hash could not be calculated," (Farmer, 2013). In our test bed, a Wireshark capture of the communication did not allow the interception of a password. Although, all other communication was unencrypted.

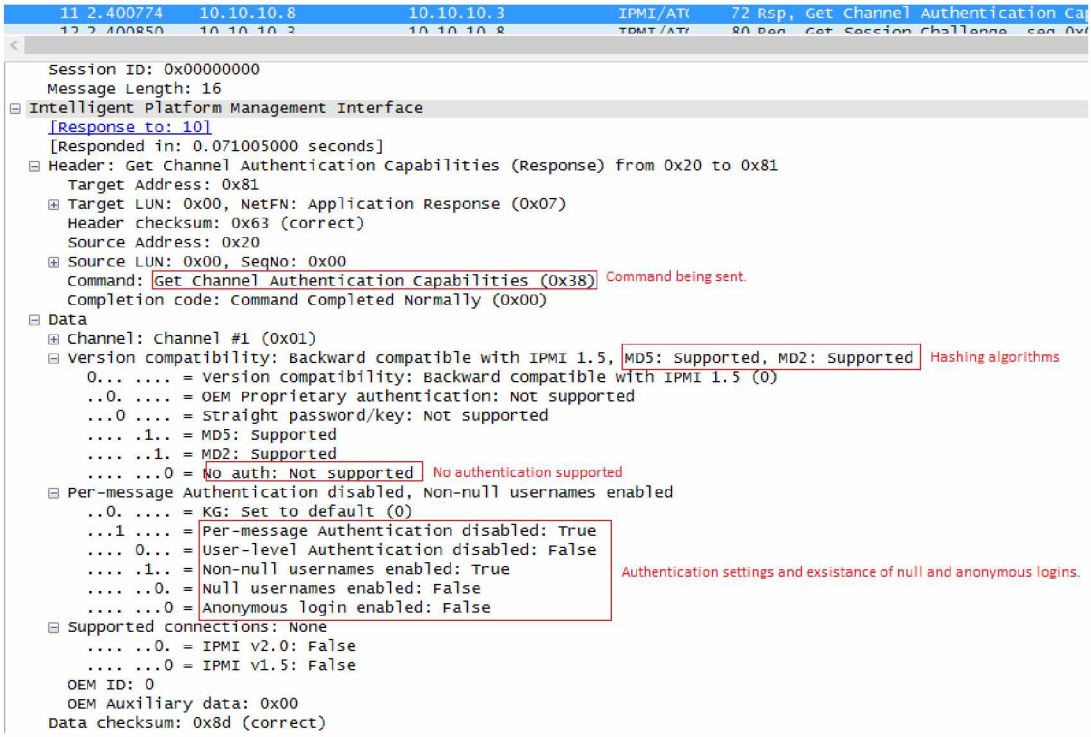
> In 1.5 to generate an AuthCode a signature is first created, using the data you'd like to authenticate, a random number sent by the BMC, a sequence number, and the signature algorithm type; this is then in turn hashed along with the password and the authentication algorithm type. The BMC may then verify that you've sent the correct data and have used the proper password. (Farmer, 2013).

Even though this deters MITM attacks or connection hijacking, a hacker can still glean useful resonance data from the unencrypted data. Such information includes the user name used to authenticate the session, the precise hashing algorithm, and the existence of users with null passwords.

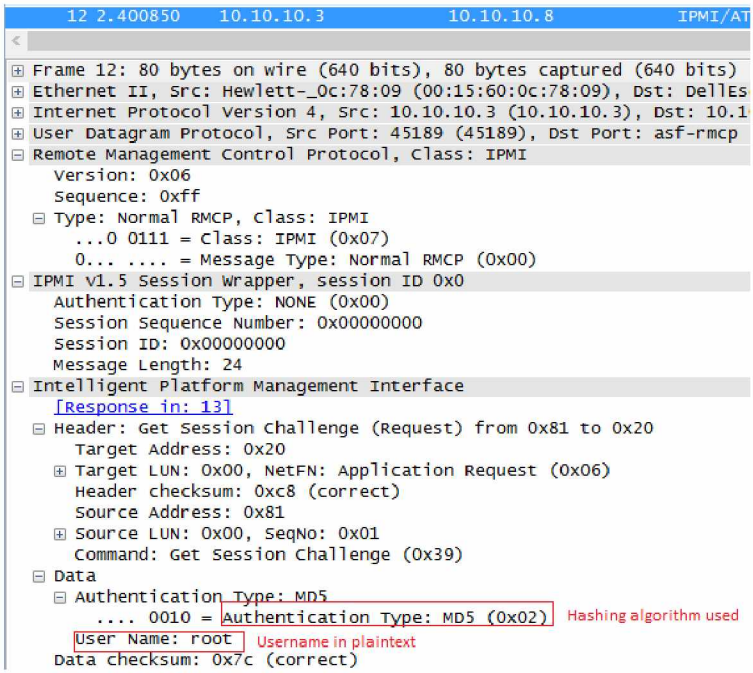
1) Here is the first packet sent from the Ubuntu server (10.10.10.3) to the iDRAC IPMI Over LAN server (10.10.10.8) where the Get Channell Authentication Capabilities command has been issued and the requested privilege level is set to Administrator:



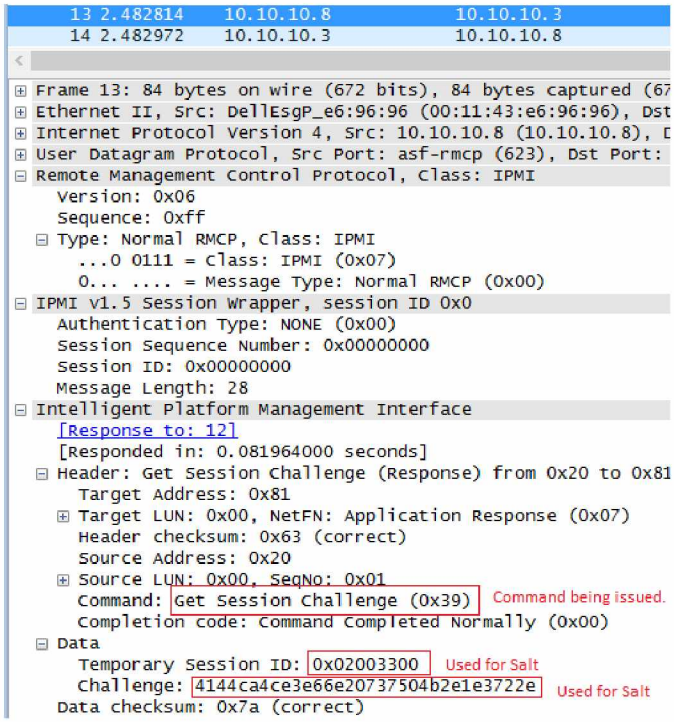

2) Here is the response packet from the iDRAC server where the hashing algorithms, authentication settings, and existence of null login credentials are stated:



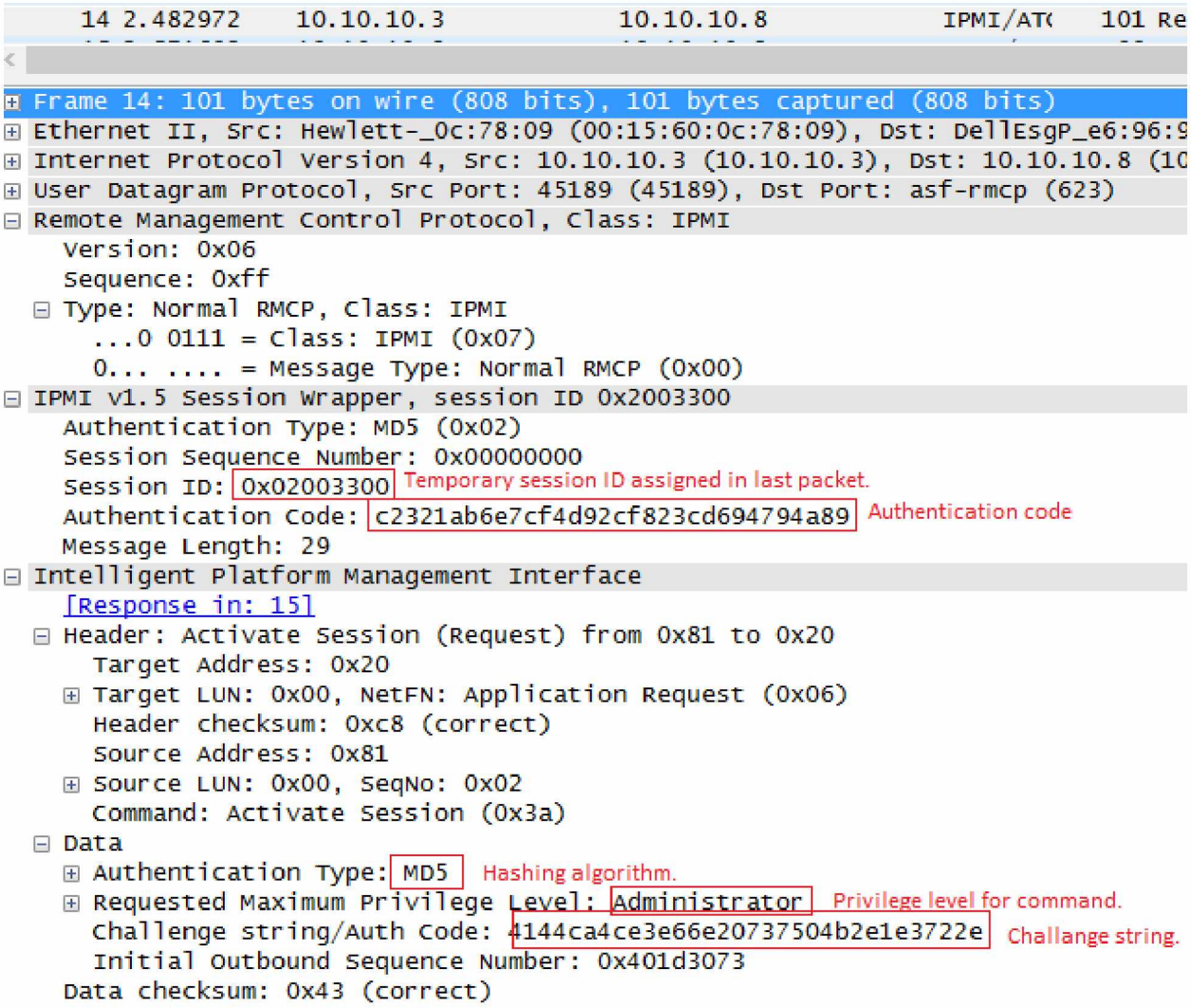
3) Here is the response from the Ubuntu Server where the hashing algorithm and user name is stated:




4) Here is the response from the iDRAC server where the Temporary Session ID and challenge used for salting the password is stated:



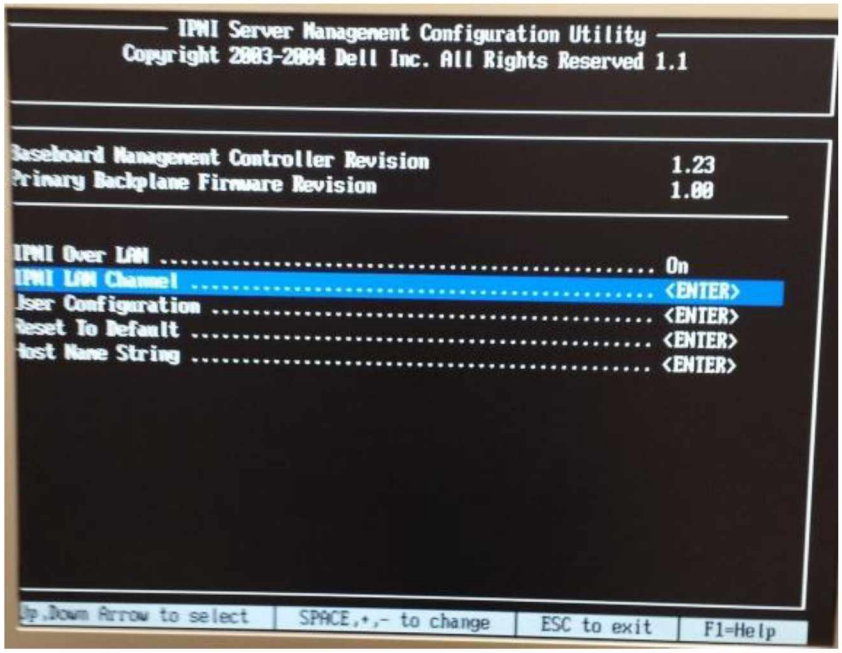
5) Here is the response from the Ubuntu server with the authentication code. The authentication code is dynamically generated and cannot be used to spoof a connection:



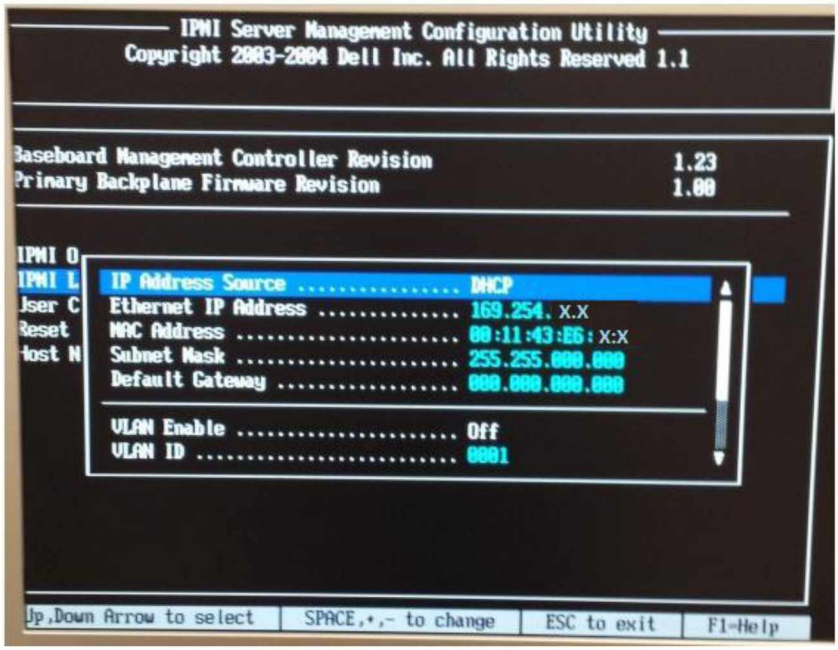
## *Enabling IPMI Over LAN*

In this section we will configure the iDRAC 4 card to enable IPMI Over LAN.

1) On POST press Ctrl+E to enter the IPMI Server Management Configuration Utility:




2) Next, go to "IPMI LAN Channel" and enter the appropriate network information:




3) Finally go to "User Configuration" and change the root password (the default is "calvin").

## *Recovering Data*

Similar to iLO, data pertaining to the previous setup can be found in the configuration menus that are shown in the section, "iDRAC Configuration." More information can be recovered by creating a new user and navigating through the iDRAC's web based interface, which is similar to iLO. The web based interface is explored in the section, "Server Decommission."

## *Server Decommission*

There is a method for gaining root access on the iDRAC 5 file system by employing the use of the hidden Remote Access Controller Admin (RACADM) commands, which were found by inspecting the RACADM utility source (Nielsen, 2014). However, a solution does not exist for iDRAC 4. As noted in the iLO Server Decommission section, the only way to truly ensure no confidential data persists on the BMC root file system is to physically destroy the BMC. Below are instructions to clear easily accessible information:

1) Under "DRAC 4" navigate to "Logs," "System Event Log (SEL)" and click "Clear Log." The System Event Log (SEL) keeps track of battery voltage, clock settings, memory errors and BMC Intrusion detection (i.e. failed login attempts).
2) Under "DRAC 4" navigate to "Logs," "DRAC 4 Log" and click "Clear Log." The DRAC 4 Log keeps track of user login, and established session information.
3) Under "DRAC 4" navigate to "Configuration," "Network" and clear any network related information.
4) Under "DRAC 4" navigate to "Configuration," "Alerts" and remove all alerts.
5) Under "DRAC 4" navigate to "Configuration," "Users" and remove all users. The Users section stores user information and their associated access privileges.
6) Under "DRAC 4" navigate to "Configuration," "Security" and "Generate a new Certificate Signing Request (CSR)" to overwrite the old one.

7) Under "DRAC 4" navigate to "Configuration," "Active Directory," remove any domain name information and overwrite the old certificate.

## *Building a Metasploit Module*

Metasploit is a powerful framework used by penetration testers to automate data reconnaissance and exploitation. In this section, we investigate how to install the Metasploit framework, how to build a basic module and, finally, how to create a more advanced scanner module to detect iLO and iDRAC related services.

### *Metasploit Set Up*

First we need to set up the Metasploit framework on the dedicated server.

1) First install the dependencies:

```
sudo apt-get install ruby1.9.1 build-essential
```

2) Download the installer script:

```
wget http://downloads.metasploit.com/data/releases/metasploit-
latest-linux-x64-installer.run
```

3) Next, give execute permissions to the file:

```
chmod +x metasploit-latest-linux-x64-installer.run
```

4) Now, run the installer:

```
sudo ./metasploit-latest-linux-x64-installer.run
```

5) Start the Metasploit service:

```
sudo /etc/init.d/metasploit start
```

6) Now we can launch Metasploit:

```
sudo msfconsole
```

## *Creating a New Module*

Now that we have the Metasploit framework setup, we will create a simple "Hello World" module to get familiar with Metasploits' modular based framework. The source code for this module can be found in the Appendix section under Source Code 1 – hello_world.rb.

1) First navigate to exploits folder:

```
cd /opt/metasploit/apps/pro/msf3/modules/exploits
```

2) Navigate to targeted operating system.

3) Create a new folder for targeted service:

```
mkdir <SERVICE_NAME>
```

4) Navigate into folder and create a new ruby script for the exploit module:

```
touch <EXPLOIT_NAME>.rb
```

5) Open up the script in your favorite editor.

6) Load the core library:

```
require 'msf/core'
```

7) Define the class:

```
class Metasploit3 < Msf::Auxiliary
```

8) Initialize the main section:

```
  def initialize
    super(
      'Name'        => 'Hello World',
      'Description' => 'Simple Hello World module.',
      'Author'      => [ 'Syler Clayton
<syler_clayton[at]hotmail.com>'],
      'License'     => MSF_LICENSE,
      'References'  =>
        [
          ['URL', 'Reference 1'],
          ['URL', 'Reference 2']
        ]
    )
  end
```

9) Now define the main function:

```
    def run
      print_good("Hello World!")
    end
  end
```

10) Now save the module.

11) Launch Metasploit:

```
sudo msfconsole
```

12) Reload modules:

```
reload_all
```

13) Now load the module:

```
use exploits/<SERVICE_NAME>/<EXPLOIT_NAME>
```

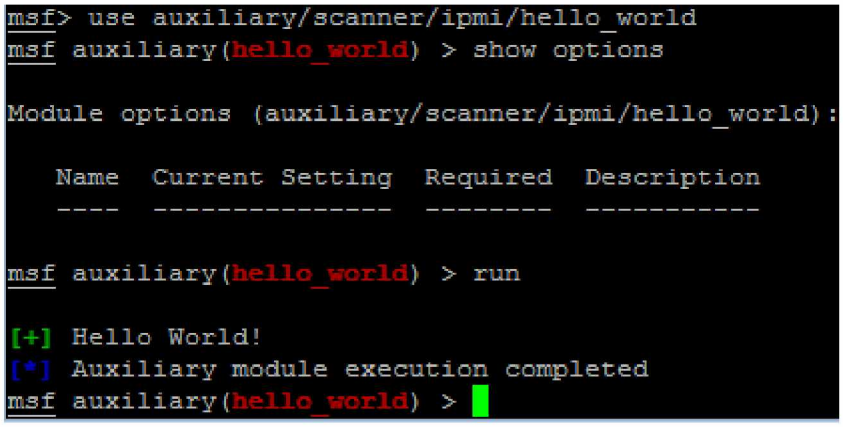
14) Show the options for the module:

```
show options
```

15) Run the exploit:

```
run
```

16) Below is a screenshot of the module running:



## *Scanner Module to Detect iLO and iDRAC Services*

H.D. Moore has done previous work writing a Metasploit module to detect and exploit IPMI Over LAN (Moore, 2013). However, a module for detecting BMC related services was non-existent. Now that we have a basic understanding of the Metasploit framework and have created a simple "Hello World" module, we can create a more advanced module to scan and detect iLO and iDRAC related services. The module works by calling nmap with verbose reporting and operating system fingerprint flags set to look for iLO and iDRAC related services. For the command `nmap -sV -F -O <HOST>` the –sV flag enables nmap version detection, the –F flag scans only ports that are named in the services file, and the –O flag triggers OS detection (Nmap, 2014). The resulting output is parsed into hardware, operating system, and related services categories using regular expressions in grep. The regular expressions look for key identifiers related to iDRAC and iLO that is given by setting the OS detection flag in nmap. Key identifiers include the strings "Dell," "iDRAC," "HP," and "iLO." The OS detection flag works by looking at the MAC address combined with fingerprinting running services. Fingerprints of known operating systems are referred to as, "reference fingerprints," and the fingerprint identified by the nmap scan is known as a "subject fingerprint" (Nmap, 2014). The subject fingerprint of the MAC address and open/closed UDP/TCP ports along with other metadata is stored and compared against the nmap-os-db database to identify the operating system and service information (Nmap, 2014). Each service is split into "Port," "State," "Service" and "Version" categories and displayed to the user. This module was 100% effective in identifying both iLO and iDRAC BMC's in our test bed (see steps 10 and 11 below) as well as correctly reporting, "No iLO or iDRAC related services found," for non-BMC hosts (see step 9 below). An example of formatted output to the user can be found below in steps 9, 10, and 11.

To install the module:

1) Run the commands:

```
cd
/opt/metasploit/apps/pro/msf3/modules/auxiliary/scanner/ipmi
sudo nano ipmi_service_scanner.rb
```

2) Copy the contents of the source code included in the "Appendix" section under "Source Code 2".

3) Launch Metasploit:

```
sudo msfconsole
```

4) Reload modules:

```
reload_all
```

5) Now load the module:

```
use auxiliary/scanner/ipmi/ipmi_service_scanner.rb
```

6) Show the options for the module:

```
show options
```

7) Set the target host IP address:

```
set RHOSTS <IP_ADDR>
```

8) Run the scanner:

```
run
```

9) Example of the scanner running against a non BMC host:

```
msf> use auxiliary/scanner/ipmi/ipmi_service_scanner
msf auxiliary(ipmi_service_scanner) > set RHOSTS 10.10.10.4
RHOSTS => 10.10.10.4
msf auxiliary(ipmi_service_scanner) > run

[+] Scanning. Please wait...
[+] No iLO or iDRAC related services found
[+] Scan complete.
[*] Auxiliary module execution completed
```

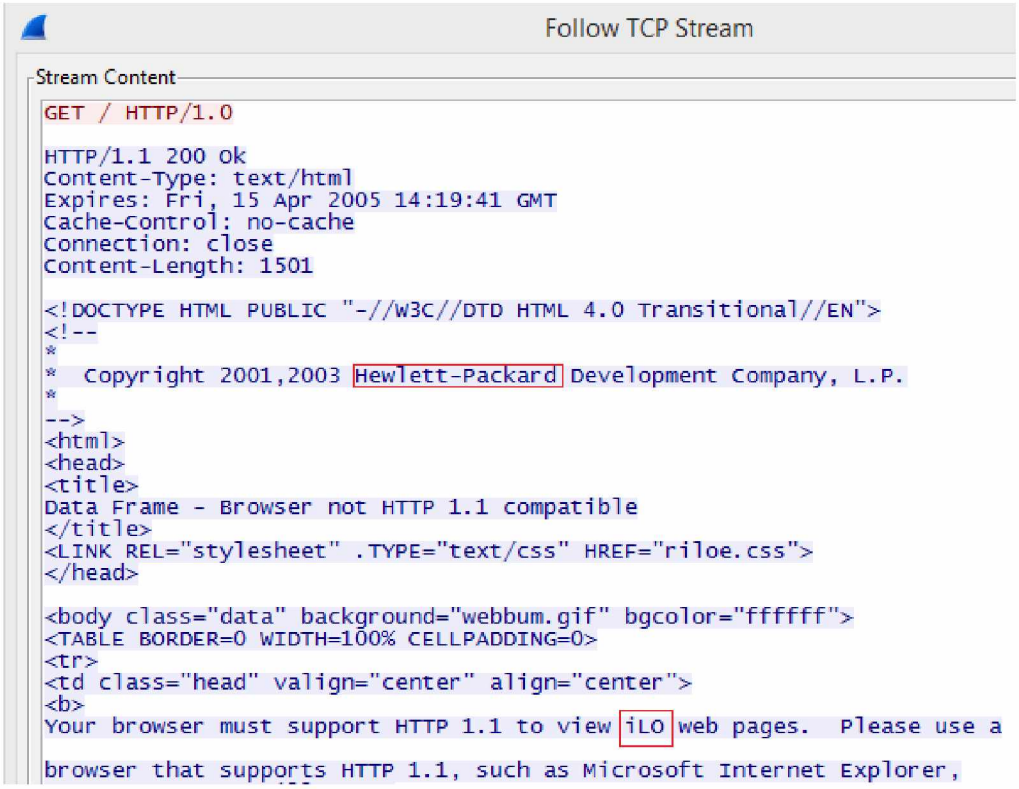
10) Example of the scanner running against a iLO host:

```
msf auxiliary(ipmi_service_scanner) > set RHOSTS 10.10.10.5
RHOSTS => 10.10.10.5
msf auxiliary(ipmi_service_scanner) > run

[+] Scanning. Please wait...
[+] iLO related services found on 10.10.10.5
[+] OS details: HP iLO or iLO 2 remote management interface

[+] PORT     STATE SERVICE    VERSION
[+] 22/tcp   open     ssh           HP Integrated Lights-Out mpSSH 0.0.1 (protoc
ol 2.0)
23/tcp   open     telnet        HP Integrated Lights-Out remote configuration te
lnetd
80/tcp   open     http          HP Remote Lights-Out http config
443/tcp  open     ssl/http      HP Remote Lights-Out http config
3389/tcp filtered ms-wbt-server

[+] Scan complete.
[*] Auxiliary module execution completed
```

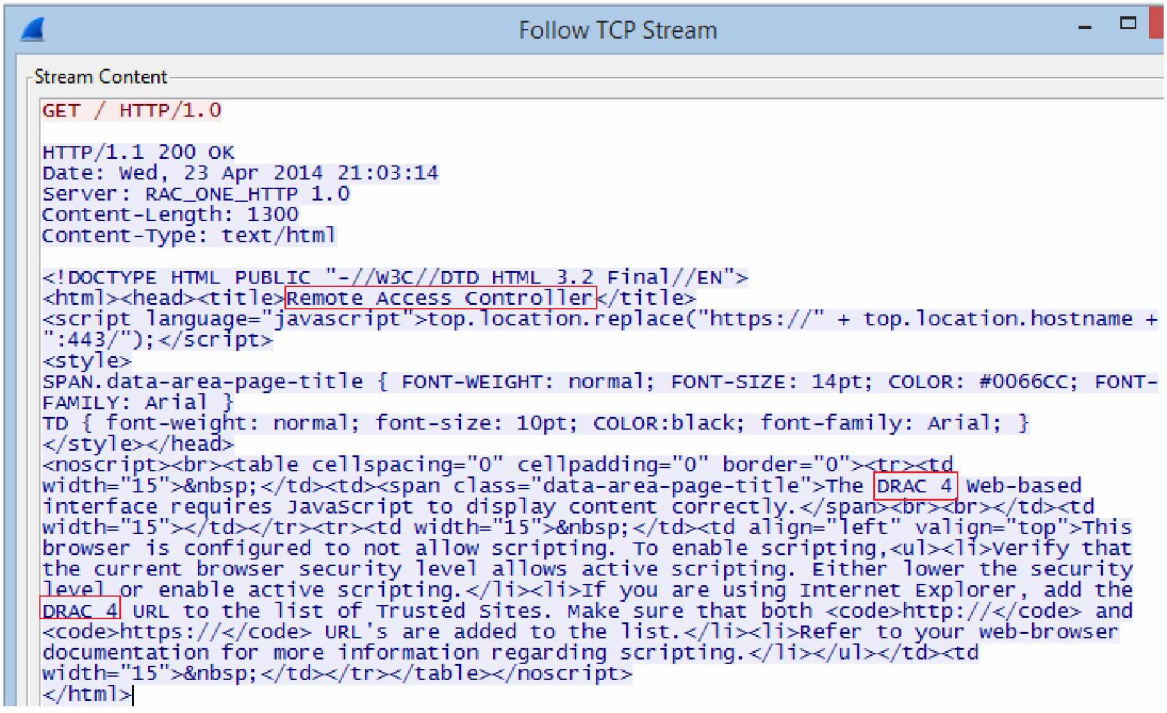
11) Example of the scanner running against a iDRAC host:

```
msf auxiliary(ipmi_service_scanner) > set RHOSTS 10.10.10.7
RHOSTS => 10.10.10.7
msf auxiliary(ipmi_service_scanner) > run

[+] Scanning. Please wait...
[+] iDRAC related services found on 10.10.10.7
[+] OS details: Dell Remote Access Controller 4 (DRAC 4)

[+] PORT     STATE SERVICE    VERSION
[+] 22/tcp   open  ssh          Mocana embedded SSH (protocol 2.0)
80/tcp   open  http         Dell Embedded Remote Access card httpd 1.0
443/tcp  open  ssl/https?
5900/tcp open  drac-console Dell Remote Access Controller 4 console

[+] Scan complete.
[*] Auxiliary module execution completed
```

12) Example of HP keywords showing up in network traffic during the nmap scan:




13) Example of Dell keywords showing up in network traffic during the nmap scan:



## *Impacts on UA Environment*

A few weeks after my first presentation on IPMI security, I received a response from Kathleen Boyle, Chief Security Officer of UAFs Office of Information Technology (OIT). She wanted to know the IP addresses, make and model of servers, and any identifying information stored on them. She stated that she would, "like to track down the groups who surpluses them to make them aware," (K. Boyle, personal communication, November 21, 2014). I responded with the information she requested. She concluded, "you picked a very good topic… we'd also be interested in any suggestions you have on how to sanitize these configs when the servers are decommissioned," (K. Boyle, personal communication, November 21, 2014). A week later, Dr. Hay told me that she had sent out a statewide email to the University of Alaska (UA) network informing staff to be aware to wipe IPMI configuration before sending equipment to surplus. A few weeks later, I received an email asking me to present at the, "OIT Distributed Technical Summit," on March 17, 2014, which I accepted.

# *Conclusions*

In conclusion, IPMI is a powerful utility for administrators. However, with great power comes great responsibility to ensure the security of the system lest control of the system fall into unauthorized hands. There are security vulnerabilities in the IPMI protocol and the various vendor implementations that need to be addressed. Best practices to avoid unauthorized access include the following:

- IPMI interfaces should only be exposed on a private network.
- IPMI software should be updated routinely.
- Strong Encryption should be enabled.
- Cipher 0 should be DISABLED.
- Accounts should follow a strong password policy.
- Accounts and configuration should be wiped before server decommission.

Depending on the vendor, BMC firmware is either publically available or only available to users with a service contract. IBM and Oracle both require service contracts (Farmer, 2013). Vendors typically abandon providing firmware and security updates to old BMC hardware. Therefore, a call for an open standard of IPMI supported hardware and firmware is needed to give users the ability to continue the support of outdated hardware similar to that of open source router firmware projects. This means that vendors need to be more transparent with their implementations and provide documentation and source code to their end users. More importantly, the IPMI specification needs to be revised to address issues directly related to the protocol such as Cipher 0 (passwordless authentication), the ability to dump hashed passwords and the storage of unencrypted passwords on the hardware (Moore, 2013). The IPMI specification is maintained by Intel and has not seen a major release since 2004, with minor revisions dating February 11, 2014 (Farmer, 2013). However, none of the revisions have formally addressed the Cipher 0 and IPMI 2.0 RAKP Authentication Remote Password Hash Retrieval Vulnerability. Finally, we

hope we have made a convincing argument that IPMI security is a recent, relevant, and widespread issue. Awareness and best practices are needed.

## *Future Work*

The scope of this project was just the tip of the IPMI security related issue iceberg. There are a multitude of IPMI supported hardware and services by vendors we did not discuss in this paper. Further investigation of these other vendors is needed:

1) Supermicro – Supermicro Intelligent Management.

2) IBM/Lenovo – Integrated Management Module (IMM).

3) Fujitsu – Integrated Remote Management Controller (iRMC).

4) Oracle - Integrated Lights Out Manager (ILOM).

Finally, I plan to give another presentation similar to the one at the OIT Distributed Tech Summit to InfraGard on August 5th, 2014. InfraGard is a partnership between the FBI and private sectors.

# *References*

Bonkoski, A., Bielawski, R., & Halderman, A. (2013). Illuminating the Security Issues Surrounding Lights-Out Server Management. *Proceedings of the 7th USENIX Workshop on Offensive Technologies, 13.* Retrieved May 1, 2014, from https://jhalderm.com/pub/papers/ipmi-woot13.pdf

Farmer, D. (2013, August 22). IPMI: FREIGHT TRAIN TO HELL. *All the IPMI that's fit to print.* Retrieved May 1, 2014, from http://www.fish2.com/ipmi/itrain.pdf

Moore, H. (2013, July 2). A Penetration Tester's Guide to IPMI and BMCs. *A Penetration Tester's Guide to IPMI and BMCs.* Retrieved May 1, 2014, from https://community.rapid7.com/community/metasploit/blog/2013/07/02/a-penetration-testers-guide-to-ipmi

Moore, H. (2013, November 6). Supermicro IPMI Firmware Vulnerabilities. *Supermicro IPMI Firmware Vulnerabilities.* Retrieved May 1, 2014, from https://community.rapid7.com/community/metasploit/blog/2013/11/06/supermicro-ipmi-firmware-vulnerabilities

Nielsen, A. (n.d.). DRAC firmware source code available for download. *Linux on Dell PowerEdgen Server.* Retrieved May 1, 2014, from http://comments.gmane.org/gmane.linux.hardware.dell.poweredge/37049

Nmap. (n.d.). Understanding an Nmap Fingerprint. *Understanding an Nmap Fingerprint.* Retrieved May 1, 2014, from http://nmap.org/book/osdetect-fingerprint-format.html

Vazquez, J. (2013, November 17). Exploiting the Supermicro Onboard IPMI Controller. *Exploiting the Supermicro Onboard IPMI Controller.* Retrieved May 1, 2014, from https://community.rapid7.com/community/metasploit/blog/2013/11/15/exploiting-the-supermicro-onboard-ipmi-controller

# *Appendix*

This section contains a glossary, a revision history and a source code section for the IPMI Server Scanner Metasploit module.

## *Glossary*

- **IPMI** – Intelligent Platform Management Interface
- **BMC** – Baseboard Management Controller
- **iDRAC** – Integrated Dell Remote Access Card
- **iLO** – Integrated Lights Out Interface
- **Cipher 0** – Allows passwordless authentication.
- **Metasploit** – Penetration testing framework.

## *Revision History*

| Version | Date | Notes |
|---|---|---|
| 1.0 | November 5, 2013 | Basic outline and documentation of previous work. |
| 1.1 | November 11, 2013 | iLO configuration and network diagram. |
| 1.2 | November 20, 2013 | Added BMC Setup section. |
| 1.3 | December 1, 2013 | Added Recovering Data section. |
| 1.4 | January 20, 2014 | Added Clearing Settings for Decommission and Configuration sections. |
| 1.5 | January 28, 2014 | Updated Clearing Settings section. |

| | | |
|---|---|---|
| 1.6 | February 8, 2014 | Formatted previous work to project write up template. |
| 1.7 | February 12, 2014 | Added Abstract and Acknowledgements sections. |
| 1.8 | February 15, 2014 | Added Introduction section. |
| 1.9 | February 22, 2014 | Added Metasploit section. |
| 2.0 | February 28, 2014 | Documented Metasploit module |
| 2.1 | March 1, 2014 | Added IPMI Background and Related work sections. |
| 2.2 | March 15, 2014 | Revised IPMI Security Threats subsections. |
| 2.3 | March 23, 2014 | Added source code for iLO scanner. |
| 2.4 | March 25, 2014 | Expanded Introduction section to reflect progress. Wrote IPMI Security Threats section. |
| 2.5 | March 27, 2014 | Added iDRAC research section (Stretch goal met). |
| 2.6 | March 30, 2014 | Wrote Conclusions section. |
| 2.7 | April 2, 2014 | Updated sources and revised entire document. |
| 2.8 | April 4, 2014 | Added Project Goals section. |
| 2.9 | April 7, 2014 | Added Future Work section. |
| 3.0 | April 12, 2014 | Added revisions based on feedback from Brandon Marken. |

| | | |
|---|---|---|
| 3.1 | April 15, 2014 | Added Example of Misconfiguration and Impact on UA Environment sections. |
| 3.2 | April 16, 2014 | Redacted IP addresses. Created BMC Diagram. Added partial revisions based on feedback from Dr. Brian Hay. |
| 3.3 | April 17, 2014 | Reworked first few sections based on feedback from Dr. Brian Hay. |
| 3.4 | April 18, 2014 | Added Glossary section. Reorganized sections and formatted Table of Contents. Reworked remaining sections based on feedback from Dr. Brian Hay. |
| 3.6 | April 21, 2014 | Changed font for commands/code to Courier New 14. Redacted UAF email. Clarified file system access in IPMI protocol section. Addressed Related Work citations and reviewed entire document for citation issues. Gave lifetime of BMC in Vendor Implementation section. Addressed file system access in iLO Recovering Data and Server Decommission sections. Expanded iLO and iDRAC server decommission sections to explain recoverable data. Addressed file system access in the iDRAC Server Decommission section. Added introduction to Creating a New Module section. Added further investigation of IPMI 1.5 authentication in the Enabling IPMI Over LAN section. Addressed IPMI protocol specification in Conclusions section. Added demonstrations of Metasploit modules. Added source for Hello World module in Appendix. Added detail to Metasploit module service scanner methods. |

| | | |
|---|---|---|
| 3.7 | April 22, 2014 | Changed code/comments font to Courier New. Added revisions in IPMI Over LAN and Scanner Module to Detect iLO and iDRAC Services sections. |
| 3.8 | April 22, 2014 | Added a pcap analysis of the IPMI over LAN 1.5 authentication process for iDRAC 4 and discussed the implications of this method in greater detail. Provided further analysis of how nmap detects BMCs. |
| 3.9 | April 23, 2014 | Changed Enabling IPMI Over LAN to IPMI Over LAN. Created new section titled Enabling IPMI Over LAN. Provided more description in the IPMI Over LAN example. Fixed off by one error in the steps describing the Metasploit module. Added screenshots of key identifiers in the Metasploit module steps. |
| 4.0 | April 23, 2014 | Received feedback from Dr. Kara Nance. Changed font to Times New Roman. Had Angela Peterson proof read and edit paper. Fixed citations to conform to the MLA format. |
| 4.1 | May 1, 2014 | Received feedback from Victoria Avery at the Writing Center. Changed format to APA. Fixed grammatical errors. |

**Table 1 - Revision History**

## *Source Code*

**Source Code 1 – hello_world.rb**

```
require 'msf/core'

class Metasploit3 < Msf::Auxiliary

  def initialize

    super(

      'Name'        => 'Hello World',

      'Description' => 'Simple Hello World module.',

      'Author'      => [ 'Syler Clayton
<syler_clayton[at]hotmail.com>'],

      'License'     => MSF_LICENSE,

      'References'  =>

        [

          ['URL', 'Reference 1'],

          ['URL', 'Reference 2']

        ]

    )

  end

  def run

    print_good("Hello World!")

  end

end
```

**Source Code 2 – ipmi_service_scanner.rb**

```
##

# This module requires Metasploit: http//metasploit.com/download

# Current source: https://github.com/rapid7/metasploit-framework

##

require 'msf/core'

class Metasploit3 < Msf::Auxiliary

  include Msf::Auxiliary::Nmap

  def initialize

    super(

      'Name'        => 'iLO v1 and iDRAC4 IPMI Information Discovery',

      'Description' => 'Discover host information through nmap
probes.',

      'Author'      => [ 'Syler Clayton <swclayton3[at]alaska.edu>'],

      'License'     => MSF_LICENSE,

      'References'  =>

        [

          ['URL', 'http://fish2.com/ipmi/'],

          ['URL',
'https://community.rapid7.com/community/metasploit/blog/2013/07/02/a-
penetration-testers-guide-to-ipmi']

        ]

    )

  end

  def run

    print_good("Scanning. Please wait...")

    rhosts = datastore['RHOSTS']

    nmapResults = %x<nmap -sV -F -O #{rhosts}>
```

```
    nmapFilter = %x<echo '#{nmapResults}' | grep -o [0-9][0-9]*/.*>
    checkDell = %x<echo '#{nmapFilter}' | grep -o "Dell">
    checkiLO = %x<echo '#{nmapFilter}' | grep -o "HP">
    checkOS = %x<echo '#{nmapResults}' | grep -o "OS details: ".*>
    if checkDell != ""
      print_good("iDRAC related services found on "+rhosts)
      print_good(checkOS)
      print_good("PORT     STATE SERVICE    VERSION")
      print_good(nmapFilter)
    elsif checkiLO != ""
      print_good("iLO related services found on "+rhosts)
      print_good(checkOS)
      print_good("PORT     STATE SERVICE    VERSION")
      print_good(nmapFilter)
    else
      print_good("No iLO or iDRAC related services found")
    end
    print_good("Scan complete.")
  end
end
```